This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A multiple processor computer system comprising:

a plurality of processors;

a shared resource;

a main memory in communication with the plurality of processors, at least a portion thereof comprising a control structure for controlling a lock on said shared resource; and

a crossbar structure connecting the plurality of processors to the shared resource for controlling access among the processors to the shared resource, the crossbar structure comprising, for each processor, a corresponding storage location, one of the plurality of processors writing to a storage location corresponding to the one processor, an address of the lock control structure associated with said shared resource to acquire the lock thereto, the crossbar structure, on behalf of the one processor, performing memory operations on the lock control structure at the address specified in the corresponding storage location in order to acquire the lock on behalf of the one processor.

2. (Original) The multiple processor computer system of claim 1 wherein the crossbar structure further comprises a second storage location corresponding to a respective processor in the multi-processor system, one of the plurality of processors writing to the corresponding second storage location an address of the lock control structure associated with said shared resource to release the lock thereto, the crossbar structure, on behalf of the processor, performing memory operations on the lock control structure at the address specified in the corresponding second storage location in order to release the lock on behalf of the processor.

3. (Original) The multiple processor computer system of claim 2 wherein the crossbar structure further comprises a third storage location corresponding to a respective processor in the multi-processor system for storing a current interrupt priority level associated with the processor wherein the current interrupt priority level represents the level of interrupt that the processor will service while spinning on a lock.

4. (Original) The multiple processor computer system of claim 3 wherein the crossbar structure further comprises a fourth storage location corresponding to a respective processor in the multi-processor system for storing a future interrupt priority level associated with the processor wherein the future interrupt priority level represents the level of interrupt that the processor will operate at upon receiving a lock.

5. (Original) The multiple processor computer system of claim 1 wherein the crossbar structure further comprises a queue for determining which processor is granted a lock.

6. (Previously Presented) The multiple processor computer system of claim 5 wherein the queue comprises a plurality of entries, each entry comprising a first data field for containing a processor identification, a second data field for containing a lock request time, and a third data field for containing a priority of a request for a lock.

7. (Previously Presented) The multiple processor computer system of claim 6 wherein each request for a lock on the shared resource has a corresponding entry in the queue, and wherein the crossbar structure determines which request is granted the lock based at least in part upon the respective priorities contained in the third data fields of the queue entries for each request.

8. (Original) The multiple processor computer system of claim 1 further comprising a multi-processor module.

9. (Original) The multiple processor computer system of claim 8 wherein the multi-processor module comprises at least two of the plurality of processors.

10. (Original) The multiple processor computer system of claim 1 wherein the shared resource comprises at least a portion of memory.

11. (Original) The multiple processor computer system of claim 1 wherein the shared resource comprises an I/O device.

12. (Original) The multiple processor computer system of claim 1 wherein the shared resource comprises a register.

13. (Original) A crossbar structure for use in a multi-processor computer system to connect a plurality of processors to at least one shared resource, the crossbar structure comprising:

for each processor, a corresponding storage location for receiving from the respective processor a memory address of a lock control structure associated with said shared resource;

wherein to acquire a lock thereto, the crossbar structure, on behalf of a processor, performs memory operations on the lock control structure at the address specified in the corresponding storage location in order to acquire the lock on behalf of the processor.

14. (Original) The crossbar structure of claim 13 further comprising:

for each processor, a corresponding second storage location for receiving from the respective processor a memory address of a lock control structure associated with said shared resource;

wherein to release a lock thereto, the crossbar structure, on behalf of a processor, performs memory operations on the lock control structure at the address specified in the corresponding second storage location in order to release the lock on behalf of the processor.

15. (Original) The crossbar structure of claim 14 further comprising:

for each processor, a corresponding third storage location for receiving from the respective processor a current interrupt priority level associated with the processor wherein the current interrupt priority level represents the level of interrupt that the processor will service while spinning on a lock.

16. (Original) The crossbar structure of claim 15 further comprising:

for each processor, a corresponding fourth storage location for storing a future interrupt priority level associated with the processor wherein the future interrupt priority level represents the level of interrupt that the processor will operate upon receiving a lock

17. (Original) The crossbar structure of claim 13 further comprising a queue for determining which processor is granted a lock.

18. (Previously Presented) The crossbar structure of claim 17 wherein the queue comprises a plurality of entries, each entry comprising a first data field for containing a processor identification, a second data field for containing a lock request time, and a third data field for containing a priority of a request for a lock.

19. (Previously Presented) The crossbar structure of claim 18 wherein each request for a lock on the shared resource has a corresponding entry in the queue, and wherein the crossbar structure determines which request is granted the lock based at least in part upon the respective priorities contained in the third data fields of the queue entries for each request.

20. (Original) A method for acquiring a lock to a shared resource in a multiprocessor computer system, the multiprocessor computer system comprising a plurality of processors, a main memory, and a crossbar structure comprising, for each processor, a first storage location for storing a memory address of a control structure associated with said shared resource to be locked, the method comprising:

writing, by one of said processors, the memory address of said control structure to said first storage location corresponding to said one processor when said one processor needs to acquire a lock on said shared resource; and

performing, by said crossbar structure, on behalf of said one processor, a memory operation on the control structure at the memory address stored in said corresponding first storage location in an attempt to acquire a lock to said shared resource; and

returning a status of the memory operation to said one processor.

21. (Original) The method of claim 20 wherein performing a memory operation comprises performing a test and set operation.

22. (Original) The method of claim 20 further comprising storing a current interrupt

priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

23. (Original) The method of claim 20 further comprising storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

24. (Original) The method of claim 23 further comprising determining a processor from the plurality of processors to be granted a lock based on a queue.

25. (Original) The method of claim 24 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

26. (Original) The method of claim 23 wherein storing the future interrupt priority level further comprises copying a value contained in a current interrupt priority register to a future interrupt priority register prior to writing, by one of said processors, the memory address.

27. (Original) The method of claim 26 further comprising copying a value contained in a future interrupt priority register to a current interrupt priority register.

28. (Original) A method for use in a multi-processor computer system having a crossbar structure that connects a plurality of processors to at least one shared resource, the method comprising:

receiving at the crossbar from one of said processors, a memory address of a

lock control structure associated with said shared resource when the processor needs to acquire a lock thereto; and

performing at the crossbar structure, on behalf of said one processor, a memory operation on the lock control structure at the memory address received from said one processor in order to acquire the lock on behalf of said one processor.

29. (Original) The method of claim 28 further comprising storing the memory address received from said one processor in a respective storage location within the crossbar structure.

30. (Original) The method of claim 28 further comprising determining a status of the lock control structure at the memory address received from said one processor.

31. (Original) The method of claim 30 further comprising sending to said one processor the determined status of the lock control structure.

32. (Original) The method of claim 28 further comprising performing at the crossbar structure memory operations on the lock control structure at the memory address received from said one processor, continually, until an unlocked status is received.

33. (Original) The method of claim 28 wherein performing a memory operation comprises performing a test and set operation.

34. (Original) The method of claim 28 further comprising receiving a current interrupt priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

35. (Original) The method of claim 28 further comprising storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

36. (Original) The method of claim 28 further comprising determining a processor from the plurality of processors to be granted a lock based on a queue.

37. (Original) The method of claim 36 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

38. (Original) A computer-readable medium having instructions stored thereon for use in a multi-processor computer system having a crossbar structure that connects a plurality of processors to at least one shared resource, the instructions, when executed on the crossbar structure, causing the crossbar structure to perform the following:

receiving at the crossbar from one of said processors, a memory address of a lock control structure associated with said shared resource when the processor needs to acquire a lock thereto; and

performing at the crossbar structure, on behalf of said one processor, a memory operation on the lock control structure at the memory address received from said one processor in order to acquire the lock on behalf of said one processor.

39. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform storing the memory address received from said one processor in a respective storage location within the crossbar structure.

40. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform determining a status of the lock control structure at the memory address received from said one processor.

41. (Original) The computer-readable medium of claim 40 wherein the instructions further cause the crossbar to perform sending to said one processor the determined status of the lock control structure.

42. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform performing at the crossbar structure memory operations on the lock control structure at the memory address received from said one processor, continually, until an unlocked status is received.

43. (Original) The computer-readable medium of claim 38 wherein performing a memory operation comprises performing a test and set operation.

44. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform receiving a current interrupt priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

45. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

46. (Original) The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform determining a processor from the plurality of processors to be granted a lock based on a queue.

47. (Original) The computer-readable medium of claim 46 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

48. (Previously Presented) The multiple processor computer system of claim 7 wherein if the third data fields of more than one of the queue entries contains the same and highest priority, the crossbar structure further determines which request is granted the lock based upon the respective lock request times contained in the second data fields of those queue entries.

49. (Previously Presented) The crossbar structure of claim 19 wherein if the third data fields of more than one of the queue entries contains the same and highest priority, the crossbar structure further determines which request is granted the lock based upon the respective lock request times contained in the second data fields of those queue entries.

50. (Previously Presented) The method of claim 24 wherein determining a processor further comprises:

reading at least one priority field and a processor identification corresponding to each priority field, from the queue;

determining that more than one priority field contains the same and highest priority;

reading a time field corresponding to each priority field containing the same and highest priority;

determining a processor based on the processor identification corresponding the earliest time in the time fields; and

granting the lock to the determined processor corresponding to the earliest time in the time fields.

51. (Previously Presented) The method of claim 36 wherein determining a processor further comprises:

reading at least one priority field and a processor identification corresponding to each priority field, from the queue;

determining that more than one priority field contains the same and highest priority;

reading a time field corresponding to each priority field containing the same and highest priority;

determining a processor based on the processor identification corresponding the earliest time in the time fields; and

granting the lock to the determined processor corresponding to the earliest time in the time fields.

52. (Previously Presented) The method of claim 46 wherein determining a processor further comprises:

reading at least one priority field and a processor identification corresponding to each priority field, from the queue;

determining that more than one priority field contains the same and highest priority;

reading a time field corresponding to each priority field containing the same and highest priority;

determining a processor based on the processor identification corresponding the earliest time in the time fields; and

granting the lock to the determined processor corresponding to the earliest time in the time fields.